What is claimed is:

Sub A1

1. A method comprising:

storing first tuples in a first table in a database system;

storing second tuples in a second table in the database system;

partitioning the first and second tuples into plural portions;

redistributing the first and second tuples to plural nodes according to the partitioning; and

hash joining the first and second tuples to produce result tuples as the first and second tuples are being redistributed to the plural nodes.

2. The method of claim 1, further comprising:

retrieving the result tuples once the hash join is performed.

3. The method of claim 1, further comprising:

retrieving the result tuples at random.

4. The method of claim 1, hash joining the first and second tuples to produce result tuples as the first and second tuples are being redistributed to the plural nodes further comprising:

producing result tuples at one of the plural nodes; and

simultaneously producing result tuples at a second of the plural nodes.

5. The method of claim 1, wherein redistributing the first and second tuples to plural nodes comprises redistributing based on split vectors containing predefined ranges.

6. The method of claim 5, wherein partitioning the first and second tuples into plural portions comprises:

partitioning first and second tuples into hash tables in each node.

7. The method of claim 6, wherein hash joining the first and second tuples comprises:

allocating a portion of a memory to a first hash table;

allocating a second portion of the memory to a second hash table; and

hash joining first tuples in the first hash table with second tuples in the second hash table.

8. The method of claim 7, wherein hash joining the first and second tuples comprises:

determining that the portion of the memory allocated to the first hash table is full;

allocating a stable storage to the first hash table; and

storing first tuples in the stable storage.

9. The method of claim 8, further comprising:

continuing to store second tuples in the second hash table; and

hash joining second tuples in the second hash table with first tuples in the first hash table.

10. The method of claim 9, further comprising:

determining that the second portion of the memory allocated to the second hash table is full;

allocating a second stable storage to the second hash table;

storing second tuples in the second stable storage; and

hash joining second tuples in the second stable storage with first tuples in the first hash table.

11. The method of claim 10, wherein hash joining the first and second tuples comprises:

generating a third hash table once all first tuples and second tuples are redistributed to each node;

retrieving one of the first tuples from the stable storage;

hash joining the one of the first tuples with tuples in the second hash table; and

storing the one of the first tuples in the third hash table.

12. The method of claim 11, further comprising:

retrieving one of the second tuples from the second stable storage; and

hash joining the one of the second tuples with tuples in the third hash table.

13. A system comprising:

a processor;

a storage; and

instructions executable by the processor, for enabling the system to:

store first tuples in a first table;

store second tuples in a second table;

partition the first and second tuples into plural portions;

redistribute the first and second tuples to plural nodes according to the partitioning; and

hash join the first and second tuples to produce result tuples as the first and second tuples are being redistributed to the plural nodes.

14. The system of claim 13, wherein the result tuples are available once the hash join is performed

15. The system of claim 13, wherein the result tuples are available at random.

16. The system of claim 13, wherein each node comprises a memory, and wherein the instructions further partition the first and second tuples into plural portions by:

partitioning first tuples into first hash tables; and

partitioning second tuples into second hash tables, wherein the hash tables are in the memory.

17. The system of claim 16, wherein the instructions further:

allocate a portion of the memory to the first hash table;

allocate a second portion of the memory to the second hash table; and

hash join first tuples in the first hash table with second tuples in the second hash table.

18. The system of claim 17, wherein the instructions further:

determine that the portion of the memory allocated to the first hash table is full; and

store first tuples in a stable storage.

19. The system of claim 18, wherein the instructions further:

continue to store second tuples in the second hash table; and

hash join second tuples in the second hash table with first tuples in the first hash table.

20. The system of claim 19, wherein the instructions further:

determine that the second portion of the memory allocated to the second hash table is full;

allocate a second stable storage to the second hash table;

store second tuples in the second stable storage; and

hash join second tuples in the second stable storage with first tuples in the first hash table.

21. The system of claim 20, wherein the instructions further:

generate a third hash table once all first tuples and second tuples are redistributed to each node;

retrieve one of the first tuples from the stable storage;

hash join the one of the first tuples with tuples in the second hash table; and

store the one of the first tuples in the third hash table.

22. The system of claim 21, wherein the instructions further:

retrieve one of the second tuples from the second stable storage; and

hash join the one of the second tuples with tuples in the third hash table.

23. An article comprising a medium storing instructions for enabling a processor-based system to:

store first tuples in a first table in a database system;

store second tuples in a second table in the database system;

partition the first and second tuples into plural portions;

redistribute the first and second tuples to plural nodes according to the partitioning; and

hash join the first and second tuples to produce result tuples as the first and second tuples are being redistributed to the plural nodes.

24. The article of claim 23, further storing instructions for enabling a processor-based system to:

retrieving the result tuples once the hash join is performed.

25. The article of claim 24, further storing instructions for enabling a processor-based system to:

redistribute based on split vectors containing predefined ranges.

26. The article of claim 25, further storing instructions for enabling a processor-based system to:

partition first and second tuples into hash tables in each node.

27. The article of claim 26, further storing instructions for enabling a processor-based system to:

allocate a portion of a memory to a first hash table;

allocate a second portion of the memory to a second hash table; and

hash join first tuples in the first hash table with second tuples in the second hash table.

28. The article of claim 27, further storing instructions for enabling a processor-based system to:

determine that the portion of the memory allocated to the first hash table is full; and

store first tuples in a stable storage.

29. The article of claim 28, further storing instructions for enabling a processor-based system to:

continue to store second tuples in the second hash table; and

hash join second tuples in the second hash table with first tuples in the first hash table.

30. The article of claim 29, further storing instructions for enabling a processor-based system to:

determine that the second portion of the memory allocated to the second hash table is full;

allocate a second stable storage to the second hash table;

store second tuples in the second stable storage; and

hash join second tuples in the second stable storage with first tuples in the first hash table.

31. The article of claim 30, further storing instructions for enabling a processor-based system to:

generate a third hash table once all first tuples and second tuples are redistributed to each node;

retrieve one of the first tuples from the stable storage;

hash join the one of the first tuples with tuples in the second hash table; and

store the one of the first tuples in the third hash table.

32. The article of claim 31, further storing instructions for enabling a processor-based system to:

retrieve one of the second tuples from the second stable storage; and

hash join the one of the second tuples with tuples in the third hash table.

add A2